Разминируем свой код: чем искать уязвимости и дефекты безопасности

Илья Поляков
руководитель отдела анализа кода
Angara Security



$this->who();

Обо мне

- Руководитель отдела анализа кода
  - Angara Security
- Также внедрял безопасную разработку в:
  - Align Technology
  - Промсвязьбанк
  - ABBYY
- Сертификаты
  - EC-Council Certified Application Security Engineer
  - Microsoft Certified Azure Security Engineer









О чём будем говорить

- «Мины» – это не только уязвимости
- Кто и как «минирует» ваш исходный код
- Какие «мины» самые популярные
- Чем его «разминировать» системным образом на ранних стадиях цикла разработки ПО

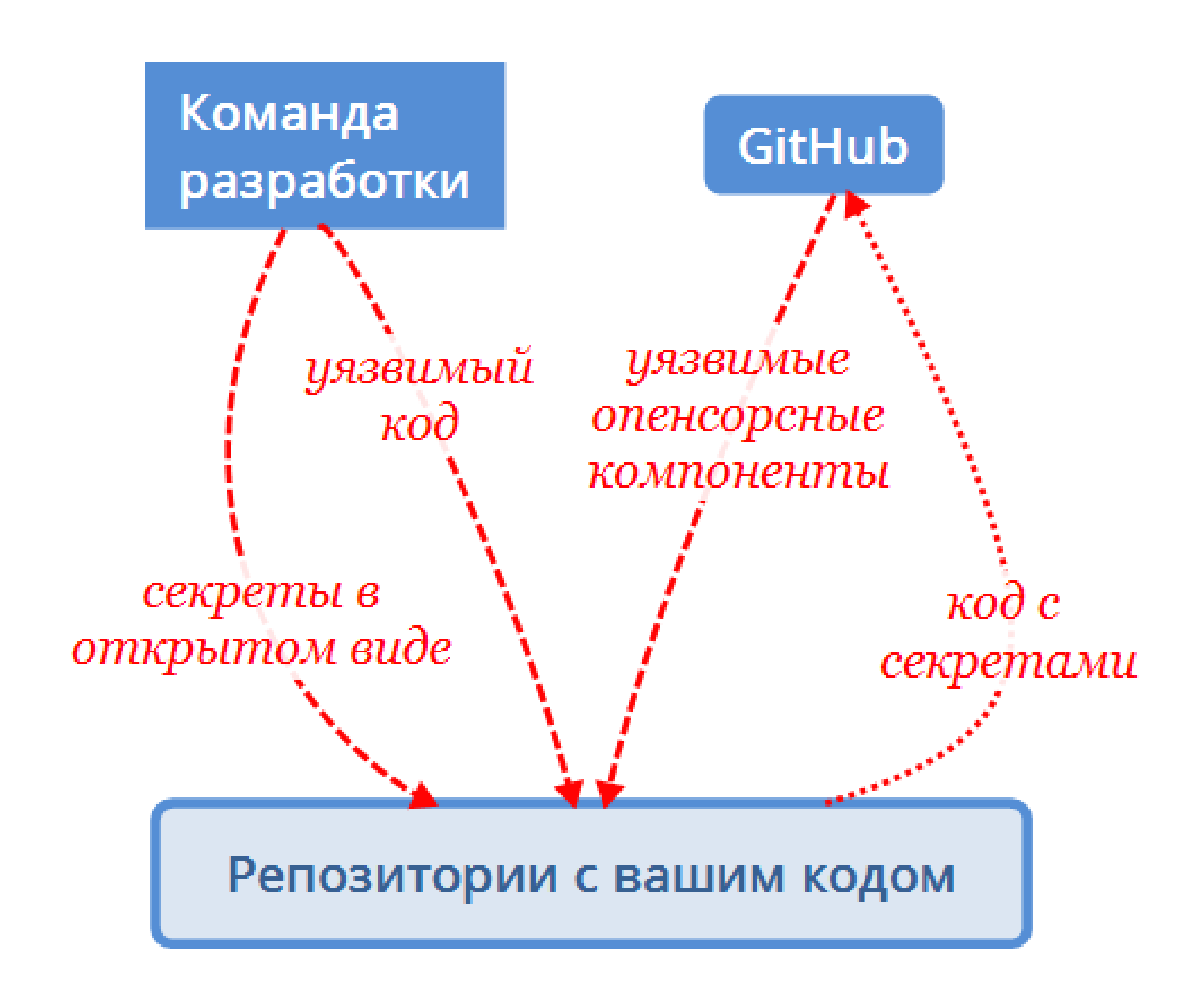


О чём НЕ будем говорить

- О динамическом анализе (DAST, IAST и прочий фаззинг)
- О сканировании контейнеров
- Об анализе IaC (Infrastructure as Code)
- О runtime-инструментах: WAF (Web Application Firewall) и RASP (Run-time Application Self-Protection)



Источники проблем в исходном коде



- Секреты: пароли системных/тестовых учёток, приватные ключи, API-токены
- Уязвимый код: ошибки разработчиков (потенциальные zeroday-уязвимости)
- Уязвимые заимствования: опенсорсные библиотеки с известными уязвимостями



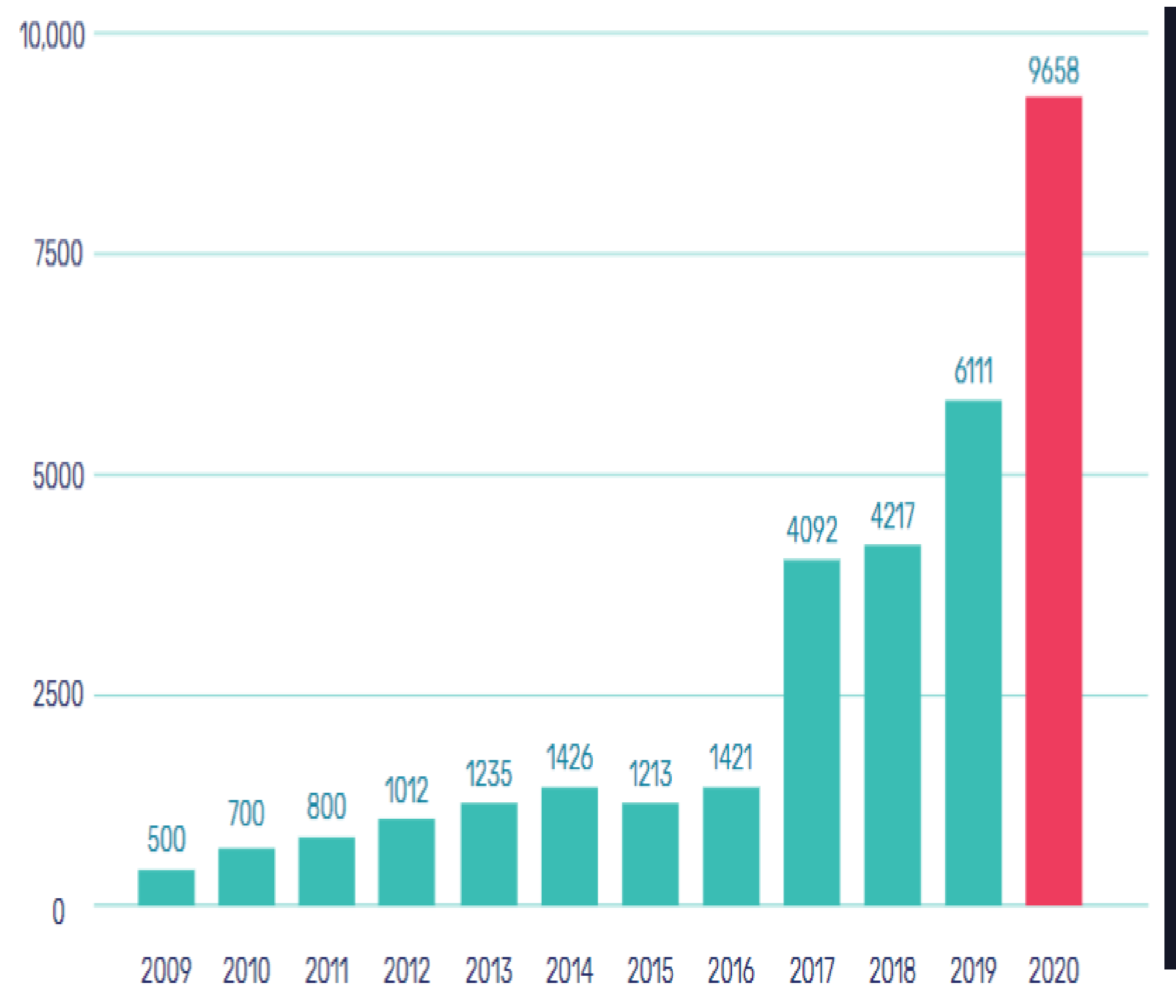
Свежий OWASP top-10 уязвимостей (все языки)

2017	2021
A01:2017-Injection	A01:2021-Broken Access Control
A02:2017-Broken Authentication	A02:2021-Cryptographic Failures
A03:2017-Sensitive Data Exposure	A03:2021-Injection
A04:2017-XML External Entities (XXE)	(New) A04:2021-Insecure Design
A05:2017-Broken Access Control	A05:2021-Security Misconfiguration
A06:2017-Security Misconfiguration	A06:2021-Vulnerable and Outdated Components
A07:2017-Cross-Site Scripting (XSS)	A07:2021-Identification and Authentication Failures
A08:2017-Insecure Deserialization	(New) A08:2021-Software and Data Integrity Failures
A09:2017-Using Components with Known Vulnerabilities	A09:2021-Security Logging and Monitoring Failures*
A10:2017-Insufficient Logging & Monitoring	(New) A10:2021-Server-Side Request Forgery (SSRF)*

* From the Survey



Open Source Vulnerabilities per Year: 2009-2020



log records
Active repos
Active Accounts
Log records
active repos
Active accounts
900M
800M
600M
400M
200M
0M
70M
60M
50M
40M
30M
20M
10M
0M
20M
15M
10M
5M
0M
2015
2016
2017
2018
2019
2020



PHP уходит в отрыв

Vulnerabilities in Top Programming Languages: 2020 vs. 2019

30%
20%
10%
0

2020
2019

GO
php
JS



Самые популярные уязвимости в опенсорсе

По отчёту Mend.io (ex-WhiteSource) за 2020 год

php	CWE-79 XSS	CWE-89 SQL Injection	CWE-352 Cross-Site Request Forgery

1	CWE-79	XSS
2	CWE-787	Out-of-bounds Write
3	CWE-125	Out-of-bounds Read
4	CWE-20	Improper Input Validation
5	CWE-200	Information Exposure
6	CWE-416	Use After Free
7	CWE-89	SQL Injection
8	CWE-22	Path Traversal
9	CWE-352	CSRF
10	CWE-190	Integer Overflow



Проблемы заимствованного кода

- Устаревшие версии с известными уязвимостями
- Пока неизвестные уязвимости (но злоумышленникам проще найти их в открытом ПО)
- Закладки
  - Компрометация учётки владельца репозитория
  - Protestware
- Лицензионный конфликт



Проблемы собственного кода

- Больше возможностей ошибиться (по сравнению с «крупноузловой» сборкой ПО)

- Исправление уязвимостей часто затягивается
  - Неочевидно, как фиксить
  - При отсутствии чёткого SLA по исправлению (надо пилить фичи!)
  - Когда уязвимость уже в проде (парадоксально, но факт)



Проблемы зашитых в код секретов

- Отучить разработчиков не хардкодить пароли непросто
- Пароли системных/тестовых учёток часто неуникальны и не меняются годами
- Даже удалённый секрет останется в истории коммитов VCS (и SAST его не найдёт)
- Код с секретами –> публичный репозиторий GitHub –> злоумышленник



Инструменты «разминирования» исходного кода

Команда разработки

GitHub

уязвимый код

уязвимые опенсорсные компоненты

SAST

секреты в открытом виде

gitleaks, SEDATED и т.д.

SCA

Репозитории с вашим кодом

- Детекторы «захардкоженных» секретов: нашёл, заблочил, поменял
- SAST: Static Application Security Testing (статический анализ)
- SCA: Software Composition Analysis (open source scanning)



Инструменты SAST (Static Application Security Testing)

- Импортные
  - Synopsys
  - Veracode
  - HCL
  - Checkmarx
- Отечественные
  - Positive Technologies
  - Ростелеком Солар
- Опенсорс
  - OWASP ASST
  - Semgrep



Инструменты SCA (Software Composition Analysis)

- Импортные
    - Synopsys
    - Snyk
    - Checkmarx
- Отечественные
    - CodeScoring
- Опенсорс
    - Dependency track
    - Dependency check



«Глубокая очистка» кода от секретов

Команда разработки

Хранилища секретов

- Опенсорсные*
  - HashiCorp Vault
  - CyberArk Conjur
- Облачные
  - Yandex Key Management Service
  - SberCloud Secret Manager

секреты

Приложения

секреты в открытом виде

gitleaks, SEDATED и т.д.

CI/CD

Репозитории с вашим кодом

* есть ограничения при бесплатном использовании



Языком антивирусов

Инструмент	Функция	Аналогия с антивирусами
• Статический анализ (SAST)	• Поиск уязвимостей в коде	• Эвристический анализ
• Компонентный анализ (SCA)	• Поиск уязвимых компонентов	• Сигнатурный анализ
	(не анализ кода!)	
• Динамический анализ (DAST)	• Анализ реального поведения	• «Песочница»



Ручной анализ кода

- Обнаружение «закладок»
- И свой, и заимствованный код
- Критические компоненты и потоки данных
- Дорого
- Человеческий фактор

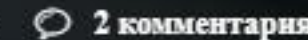
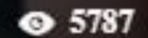




«Эту уязвимость невозможно найти снаружи»

НОВОСТИ

Компанию Uber взломали. Хакер мог украсть исходные коды и информацию об уязвимостях

Мария Нефёдова , 16.09.2022  2 комментария  5787



Сканеры кода: plug’n’pray?

- Интеграция средств безопасности в конвейер разработки – дело нехитрое
- Но нельзя просто воткнуть сканеры в пайплайн и расслабиться
- Нужен кто-то, понимающий в эксплуатации уязвимостей





Безопасность – это конкурс по копанию



критичность уязвимости

ущерб

Junior

Middle

Senior

квалификация AppSec-инженера

script kiddle

criminals

APT

тип атаки



Скупой платит дважды

- Безопасность – удовольствие не из дешёвых

- Её отсутствие ещё дороже (и чревато неудовольствием)





Голосуйте за доклад:



Илья Поляков
Telegram: @E_Liu_Ha



PHP Russia
2022